A group-theoretic approach to formalizing bootstrapping problems

Andrea Censi Richard M. Murray

***Abstract*—The bootstrapping problem consists in designing agents that learn a model of themselves and the world, and utilize it to achieve useful tasks. It is different from other learning problems as the agent starts with uninterpreted observations and commands, and with minimal prior information about the world. In this paper, we give a mathematical formalization of this aspect of the problem. We argue that the vague constraint of having "no prior information" can be recast as a precise algebraic condition on the agent: that its behavior is invariant to particular classes of nuisances on the world, which we show can be well represented by actions of groups (diffeomorphisms, permutations, linear transformations) on observations and commands. We then introduce the class of bilinear gradient dynamics sensors (BGDS) as a candidate for learning generic robotic sensorimotor cascades. We show how framing the problem as rejection of group nuisances allows a compact and modular analysis of typical preprocessing stages, such as learning the topology of the sensors. We demonstrate learning and using such models on real-world range-finder and camera data from publicly available datasets.**

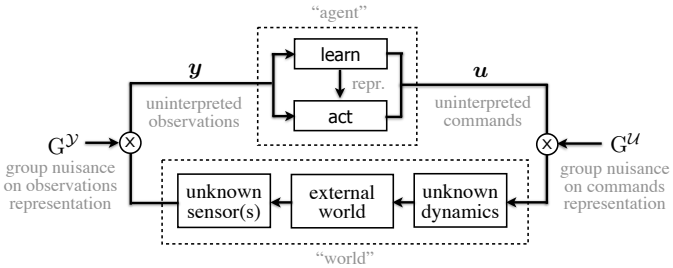


Figure 1. We consider the problem of designing agents that learn a model of the world, including their own dynamics, with "no prior information" about the system. So far, there has not been a precise formalization of this notion. In this paper, we argue that the agent not needing certain information/assumptions is equivalent to the algebraic condition of its *behavior* being *invariant* to certain classes of lossless transformations of input and output signals. Such transformations are well modeled by group actions (e.g., permutations, linear transformations, diffeomorphisms). This language allows to define exactly the limits of a bootstrapping agent in terms of the group nuisances it can tolerate.

I. Introduction

The striking difference between artificial and natural intelligent systems is that the latter rely on self-organizing cognitive processes able to process heterogenous data sources [1], while the artificial systems we can build are for the most part rigidly designed with minimal adaptive/learning components. This often results in fragile systems that cannot adapt to unexpected scenarios and unplanned changes in the models.

The idea of designing self-organizing systems has been explored recently in machine learning for tasks such as detection, clustering, and classification. It is now understood that the best way to extract useful features from the data is to use neural networks that act as universal approximators and can "learn the geometry of the data" unsupervisedly [2], [3]. Other parallel efforts share the same spirit of learning generic representations from uninterpreted data with minimal prior information [4].

Can the same approach be used in robotics, where the agent must act on the world, rather than just collecting measurements? Can we realize "plug-and-play" robotic system, where the agent learns to use any set of sensors and actuators from scratch, with no prior information about them? (Fig. 1)

Similar problems are studied in developmental/epigenetic robotics [5], [6], which concerns the development of an embodied intelligence, guided by its interaction with the real world. Kuipers and colleagues [7]–[10] provide examples of robotic agents that can learn and use their sensorimotor cascade from uninterpreted streams of observations and commands.

This approach is fundamentally different from the typical view in reinforcement learning, where the interaction of an agent with the world is modeled through a reward function [11] as those results tend to be generic and do not target the case of an agent interacting with the real world. In fact, a generic theory of learning from rewards for disembodied agents is available [12] but does not lead to interesting results for robotics. The idea of starting with "no prior information" about the world, and with an unknown state space, sets the problem apart from other uses of learning techniques in robotics that assume specific knowledge of kinematic models (e.g., [13], [14]).

Previous work: In [15], we approach bootstrapping from a control theory perspective, to address the lack of strong theoretical results on the properties of bootstrapping agents. We do this by using models simple enough to analyze, yet general enough to represent various sensors. In particular, we consider sensorimotor cascades composed by omnidirectional kinematics and three "canonical" exteroceptive sensors: field samplers[1], range-finders, and cameras. Their dynamics are quite similar, if we write the differential equations governing the raw observations (Table I). We consider the class of

The authors are with the Control & Dynamical Systems department, California Institute of Technology. E-mail: {andrea,murray}@cds.caltech.edu. Supplemental material available at http://purl.org/censi/2011/bgds.

[1] A *field sampler* is a sensor that samples a local intensity field (temperature, odor, and other generalized fields [16]).

Table I
Continuous dynamics of canonical robotic sensors

sensor	$\mathcal{S}$	*continuous dynamics (far from occlusions)*
field sampler	$\mathbb{R}^3$	$\dot{y}(s) = \nabla_i y(s) \boldsymbol{v}^i + (s \times \nabla y(s))_i \, \boldsymbol{\omega}^i$
camera	$\mathbb{S}^2$	$\dot{y}(s) = \mu(s) \nabla_i y(s) \boldsymbol{v}^i + (s \times \nabla y(s))_i \, \boldsymbol{\omega}^i$
range-finder	$\mathbb{S}^2$	$\dot{y}(s) = (\nabla_i \log y(s) - s_i^*) \boldsymbol{v}^i + (s \times \nabla y(s))_i \, \boldsymbol{\omega}^i$

Here $\boldsymbol{v} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ are linear and angular velocities; s is a continuous index ranging over the "sensel space" $\mathcal{S}$ (either the unit sphere $\mathbb{S}^2$ or $\mathbb{R}^3$); $y(s)$ is the raw sensor data (field intensity value, pixel luminance, or range reading); ∇ is spatial gradient with respect to s; $\mu(s)$ is the nearness; and repeated indices are summed over (Einstein notation).

bilinear dynamics sensors (BDS) as generic approximators for such sensorimotor cascades, and discuss an agent that can learn such models unsupervisedly. Rather than letting the agent behavior be guided by a generic reward function, we study a particular task, servoing, that makes sense for multiple sensor modalities and does not depend on the observations semantics.

Contribution: The first contribution of this paper is a formalization of the vague requirements of the agent not having "a priori information" about the world. Section II introduces the idea that the agent not having assumptions regarding a certain property of the world is equivalent to requiring that the agent behavior is *invariant* to all transformations that change that property. It is natural to talk about these transformations as *group nuisances* acting on the *representation* of dynamical systems. Note that this use of groups is related to, but fundamentally different than, their use for representing time-varying nuisances on the data [17], or as a convenient parametrization of the robot configuration (e.g., [18]). This idea provides both a conceptual framework and a falsifiable test for assessing exactly what a priori information is needed by an agent. Section III recasts the analysis in [15] with this new language.

Table II
SYMBOLS USED IN THIS PAPER

Formalization of agents-world interaction	
$\mathcal{D}(\mathcal{B},\mathcal{A})$	"Black box" dynamical systems with input in $\mathcal{A}$ and output in $\mathcal{B}$ (Definition 1).
$\mathcal{U}$	Generic commands space.
$\mathcal{Y}$	Generic observations space.
W	$\in \mathcal{D}(\mathcal{Y},\mathcal{U})$ The "world": everything between observations and commands.
$\mathrm{G}^{\mathcal{U}}$	Group nuisances acting on the commands representation.
$\mathrm{G}^{\mathcal{Y}}$	Group nuisances acting on the observations representation.
$\mathsf{Agents}(\mathcal{U},\mathcal{Y})$	Set of agents that interact with a world in $\mathcal{D}(\mathcal{Y},\mathcal{U})$ (Definition 4).
$\mathcal{R}$	An agent's representation space.
learn	$\mathcal{D}(\mathcal{U}\times\mathcal{R},\mathcal{Y})$ The agent's exploration/learning strategy.
act	$:\mathcal{R}\to\mathcal{D}(\mathcal{U},\mathcal{Y})$ The agent's action strategy.
BDS *models and agents*	
n	Number of sensels (observations)
k	Number of commands.
$\mathsf{BDS}(n,k)$	$\subset \mathcal{D}(\mathbb{R}^n,\mathbb{R}^k)$ Bilinear dynamics sensors (BDS) models with k commands and n observations (Definition 6).
$A_{\mathsf{BDS}}(k,n)$	$\subset \mathsf{Agents}(\mathbb{R}^k,\mathbb{R}^n)$ Agent designed for BDS systems (Definition 7)
BGDS *models and agents*	
$\mathcal{S}$	The sensel space.
d	Dimension of $\mathcal{S}$.
$\mathsf{C}(\mathcal{S};\mathbb{R})$	Smooth maps from $\mathcal{S}$ to $\mathbb{R}$.
$\mathsf{BGDS}(\mathcal{S},k)$	$\subset \mathcal{D}(\mathsf{C}(\mathcal{S};\mathbb{R}),\mathbb{R}^k)$ Bilinear gradient dynamics sensors (BGDS) models with k commands and observations consisting of fields on $\mathcal{S}$ (Definition 10).
$A_{\mathsf{BGDS}}(k,\mathcal{S})$	$\subset \mathsf{Agents}(\mathbb{R}^k,\mathsf{C}(\mathcal{S};\mathbb{R}))$ Agent designed for BGDS systems (Definition 12)
$\mathsf{pop}_{\psi}(\mathcal{S},\mathcal{O})$	$\in \mathcal{D}(\mathsf{C}(\mathcal{S}\times\mathcal{O};\mathbb{R}),\mathsf{C}(\mathcal{S};\mathcal{O}))$ Population code with kernel ψ (Definition 16)
$\mathsf{RF}(\mathbb{S}^m,k)$	$\subset \mathcal{D}(\mathsf{C}(\mathbb{S}^m;\mathbb{R}),\mathbb{R}^k)$ Sensorimotor cascade with kinematic inputs and range-finder sensors.

Section IV introduces the family of *bilinear gradient dynamics sensors* (BGDS) models. They are a specialization of BDS models that allows more efficient representations. However, they need the equivalent of an "intrinsic calibration". The group nuisances formalism allows to integrate in the analysis the results from existing methods for intrinsic calibration of generic data sources [10], [19]–[21], by discussing the group nuisances that they normalize or introduce. Section V discusses, with the same language, the ability of BGDS models to represent the nonlinear dynamics of a range-finder given some minimal preprocessing.

Section VI shows demonstration with range-finder and camera data. In [15] we used servoing as an example task; here we consider the problem of *anomaly detection*, consisting in detecting which changes in the observations are due to the agent motion and which to independent causes (e.g., objects that move independently from the agent). This is a passive task for which we can use publicly available logged data.

A. Notation

Basics: $\mathbb{R}_0^+$ is the set of positive reals. $\mathsf{measures}(\mathcal{B})$ the set of all probability measures on the space $\mathcal{B}$.

Groups: We use the language of basic group theory. Standard references are Rothman [22] for basic notions and Vadarajan [23] for advanced topics. $\mathsf{Diff}(\mathcal{M})$ is the set of all diffeomorphisms from $\mathcal{M}$ to itself. $\mathsf{Diff}^+(\mathcal{M})$ denotes orientation-preserving diffeomorphisms. $\mathsf{GL}(n)$, the general linear group, is the set of all linear invertible maps on $\mathbb{R}^n$ represented by $n \times n$ invertible matrices. $\mathsf{Perm}(n)$ is the permutation group on sequences of length n. $\mathsf{SE}(m)$ is the special euclidean group.

Tensors: We use basic notions from differential geometry and tensor analysis. Standard references are Do Carmo [24] for basic notions and Abraham, Marsden and Ratiu [25] for advanced topics. Throughout the paper, we use the Einstein convention that assumes summation over repeated indices up and down. For example, $a_i b^i$ is equivalent to $\sum_i a_i b^i$. $\mathbb{S}^m \subset \mathbb{R}^{m+1}$ is the unit sphere.

Other symbols that will be defined in the paper are shown in Table II for reference.

II. BOOTSTRAPPING AS REJECTION OF GROUP NUISANCES

Our intent is to formalize what it means for an agent to have "no prior information" about its body and environment; or, in general, how to formalize that only *certain* information is needed. We start from a quite abstract formalization of the world, characterized only through its input-output behavior (Section II-A). Then we show how these models are changed under certain types of transformations of the input and output signals (Section II-B). We define what we mean by a learning agent in this context (Section II-C). Then we formalize the idea of bootstrapping as rejection to group nuisances (Section II-D).

A. Modelling the world as an uncertain black box

If the agent does not know anything about either the world or its own sensorimotor cascade, the line between the two is quite blurred, at least at the beginning. We call *world* whatever lies between commands and observations, which includes both the external world and the agent's own dynamics (Fig. 1).

We use a definition of dynamical system (both for the world and the agent) that does not require a definition of the state space, and incorporates uncertainty at a fundamental level.

Definition 1. Given an ordered set $\mathbb{T}$ representing time, an input space $\mathcal{A}$ and an output space $\mathcal{B}$, a (random) dynamical system can be represented by a map $\boldsymbol{D} : \mathcal{B}^{\mathbb{T}} \times \mathcal{A}^{\mathbb{T}} \to \mathsf{measures}(\mathcal{B})$ from the history of input signal $\boldsymbol{a}_{\mathbb{T}} \in \mathcal{A}^{\mathbb{T}}$ and output signals $\boldsymbol{b}_{\mathbb{T}} \in \mathcal{B}^{\mathbb{T}}$ such that $\boldsymbol{D}(\boldsymbol{b}_{\mathbb{T}}, \boldsymbol{a}_{\mathbb{T}}) \in \mathsf{measures}(\mathcal{B})$ is the probability density of the next observations given the history.

Example 2. In the case of a discrete time stochastic system with observations $\boldsymbol{b}_{\mathbb{T}} = y_{0:t}$ and input $\boldsymbol{a}_{\mathbb{T}} = u_{0:t}$, the measure $\boldsymbol{D}(\langle \boldsymbol{b}_{\mathbb{T}}, \boldsymbol{a}_{\mathbb{T}} \rangle)$ is simply the distribution $p(y_t | y_{0:t}, u_{0:t})$, which can be written as a function of the observation model $p(y|x)$, the transition model $p(x_t|x_{t-1}, u_t)$, and the prior $p(x_0)$. Non-stochastic systems are derived as a particular case.

We denote by $\mathcal{D}(\mathcal{B}, \mathcal{A})$ the set of all dynamical systems so defined (the order $\mathcal{B}, \mathcal{A}$ will prove to be the most convenient in the following). We will need also the idea of small distortion of a model. Given a subset $\mathcal{F} \subset \mathcal{D}(\mathcal{B}, \mathcal{A})$ and $\epsilon \geq 0$, we define as $\mathcal{F} \oplus \epsilon \subset \mathcal{D}(\mathcal{B}, \mathcal{A})$ an enlargement of the family $\mathcal{F}$ by an ϵ-distortion in a metric defined on $\mathsf{measures}(\mathcal{B})$.

The definition essentially sees the system as a noisy black box[2], and does not mention an internal state space explicitly: the state is implicit in the fact that the map depends on the complete history of commands and observations. This view is very close to the epistemological perspective of an agent that starts with zero information about the world: the knowledge of the state space is inaccessible, and uncertainty has a dominant role.

B. Group actions on dynamical systems

We are interested in defining how such models can be manipulated by transformations that somehow preserve their "essence" but change their "appearance". For example, consider a system with input $\boldsymbol{u} \in \mathbb{R}^2$ and observations $\boldsymbol{y}$. Suppose the roles of the two commands are swapped: let $\boldsymbol{u} = \left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right] \boldsymbol{u}'$ and consider the dynamical system from $\boldsymbol{u}'$ to $\boldsymbol{y}$. This transformation does change the system in every conventional sense, but does not change the situation much, if the agent does not have prior information about the meaning of each command. A relabeling of k commands can be represented by a permutation $\sigma \in \mathsf{Perm}(k)$ such that $u_i = u'_{\sigma(i)}$. Permutations are a group because they can be composed and reversed. In general, every transformation of the input/output signals of a system that does not lose information must be the action of some group, because it can be reversed and composed. Therefore, our assumption is that the nuisances under investigation can be represented by groups acting either on the input signals or on the output signals of a given system. If a group acts on the input/output signals, then its action can be defined on the system as well. We can define the meaning of a mapping $\boldsymbol{D} \mapsto h \cdot \boldsymbol{D} \cdot g$ where, reading right to left, the input signals are filtered by the group element g; then the system $\boldsymbol{D}$ produces an output, which is filtered according to the group element h. The following is the (rather technical) definition compatible with Definition 1.

Definition 3. Let $\boldsymbol{D} \in \mathcal{D}(\mathcal{B}, \mathcal{A})$. Then a group $\mathrm{G}^{\mathcal{A}}$ acting on the input space $\mathcal{A}$ defines a *right action* on the model $\mathcal{D}$ mapping $\boldsymbol{D} \mapsto \boldsymbol{D} \cdot g$, where $\boldsymbol{D} \cdot g$ is defined by

$$[\boldsymbol{D} \cdot g](\boldsymbol{b}_{\mathbb{T}}, \boldsymbol{a}_{\mathbb{T}}) = \boldsymbol{D}(\boldsymbol{b}_{\mathbb{T}}, g\boldsymbol{a}_{\mathbb{T}}), \qquad g \in \mathrm{G}^{\mathcal{A}}. \tag{1}$$

Likewise, a group $\mathrm{G}^{\mathcal{B}}$ acting on the output space $\mathcal{B}$ defines a *left action* $\boldsymbol{D} \mapsto h \cdot \boldsymbol{D}$, where $h \cdot \boldsymbol{D}$ is defined by

$$[h \cdot \boldsymbol{D}](\boldsymbol{b}_{\mathbb{T}}, \boldsymbol{a}_{\mathbb{T}}) = h \cdot \boldsymbol{D}(h^{-1} \cdot \boldsymbol{b}_{\mathbb{T}}, \boldsymbol{a}_{\mathbb{T}}), \qquad h \in \mathrm{G}^{\mathcal{B}}. \tag{2}$$

C. Defining bootstrapping agents

Let $\mathcal{U}$ be the command space, $\mathcal{Y}$ be the observations space, and $\mathsf{W} \in \mathcal{D}(\mathcal{Y}, \mathcal{U})$ represent the model of the "world" (everything in between observations and commands). We define bootstrapping agents by a pair of strategies: the first strategy consists in learning some *representation* of the world, and the second uses that representation to *do* (or to *estimate*) something.

Definition 4. A bootstrapping agent for a world $\mathsf{W} \in \mathcal{D}(\mathcal{Y}, \mathcal{U})$ is a tuple $\langle \mathcal{R}, \mathsf{learn}, \mathsf{act} \rangle$ such that $\mathcal{R}$ is the representation space, $\mathsf{learn} \in \mathcal{D}(\mathcal{U} \times \mathcal{R}, \mathcal{Y})$ is the learning/exploration strategy; and $\mathsf{act} : \mathcal{R} \to \mathcal{D}(\mathcal{U}, \mathcal{Y})$ is the action/estimation phase.

We define as $\mathsf{Agents}(\mathcal{U}, \mathcal{Y})$ the set of all agents interacting with the world through commands in $\mathcal{U}$ and observations $\mathcal{Y}$.

The *learning strategy* learn is defined as an element of $\mathcal{D}(\mathcal{U} \times \mathcal{R}, \mathcal{Y})$, which means it is a dynamical system which has as input the observations ($\mathcal{Y}$), and two output signals: the commands ($\mathcal{U}$) that drive the exploration, and the internal representation ($\mathcal{R}$) which is being estimated. In this discussion we are neglecting all sorts of details about how to properly define the training phase (when to stop it; the tradeoff of exploration/exploitation; etc); those concerns are important but somewhat orthogonal to our main interest. We denote by $\mathsf{learn}(\mathsf{W}) \in \mathcal{R}$ the representation learned after a suitable training phase.

The *acting strategy* act is a map from $\mathcal{R}$ to $\mathcal{D}(\mathcal{U}, \mathcal{Y})$; this means that the learned representation $\mathcal{R}$ is converted into a dynamical system which will do the actual interaction with the world. We remark that this dynamical system has, in general, an internal state, and Definition 1 allows randomized behavior. For example, $\mathcal{R}$ might include a description of the sensor calibration and the statistics of the environment; from that, one generates the dynamical system $\mathsf{act}(\mathcal{R}) \in \mathcal{D}(\mathcal{U}, \mathcal{Y})$ which might include logic for estimation of an internal state.

D. Bootstrapping as invariance to group actions

We have defined the world, the agent, and how the world transforms under group nuisances. At this point, we can

[2]Predictive State Representation models [26] are similar in spirit.

introduce the main theoretical point of this paper: *it is possible to transform vague constraints such as "the agent has no assumptions on the world" into precise algebraic conditions on the world-agent loop; specifically, an agent does not need certain information if its behavior is invariant to group nuisances acting on the world that destroy that particular information*. The following is the formal statement.

Definition 5. Let W belong to a family of models $\mathcal{W} \subset \mathcal{D}(\mathcal{Y}, \mathcal{U})$. Let the groups $\mathrm{G}^{\mathcal{U}}$, $\mathrm{G}^{\mathcal{Y}}$ be left and right actions on W. We say that an agent $\langle \mathcal{R}, \mathsf{learn}, \mathsf{act} \rangle$ is invariant to the action of $(\mathrm{G}^{\mathcal{U}}, \mathrm{G}^{\mathcal{Y}})$ for the family $\mathcal{W}$ if

$$(\mathsf{act} \circ \mathsf{learn})(\boldsymbol{h} \cdot \mathsf{W} \cdot \boldsymbol{g}) = \boldsymbol{g}^{-1} \cdot (\mathsf{act} \circ \mathsf{learn})(\mathsf{W}) \cdot \boldsymbol{h}^{-1}$$

for all $\boldsymbol{h} \in \mathrm{G}^{\mathcal{Y}}$, $\boldsymbol{g} \in \mathrm{G}^{\mathcal{U}}$, and $\mathsf{W} \in \mathcal{W}$.

It is easy to see that, if this condition holds, then the nuisances have no effect on the agent's actions ($\boldsymbol{g}^{-1}$ and $\boldsymbol{g}$ cancel, and likewise for $\boldsymbol{h}$). The simplest example is when the groups represent linear scaling (gains of the actuators, or measurements units); if the gain is doubled, we expect that the produced commands will be halved.

While the input-output behavior is unchanged by nuisance, the internal representation is allowed to change; what happens to the internal representation is an interesting question that we will not investigate in this paper, in which we treat the representation mostly as an opaque object. Putting constraints on the *behavior* rather than the internal representation makes it clear that such constraints are immediately falsifiable, analytically or by direct observation of the agent.

III. Analysis for BDS models and agents

Now that we have a language to say exactly what we require of a bootstrapping agent, we apply it to the results in previous work, as a simple example in preparation to the new results described later. In [15] we studied this class of Bilinear Dynamical Sensor (BDS) models.

Definition 6. A system is in $\mathsf{BDS}(n, k)$ if the observations $\boldsymbol{y} \in \mathbb{R}^n$, the commands $\boldsymbol{u} \in \mathbb{R}^k$, and there exists a $(n, n \times k)$ tensor $\mathbf{M}$ such that[3] $\dot{y}^s = \mathsf{M}^s_{vi} y^v u^i$.

Here s, v are indices that span over the n sensels. Writing the system in the form $\dot{\boldsymbol{y}} = (\mathsf{M}^{\vdots}_{:1}\boldsymbol{y})u^1 + (\mathsf{M}^{\vdots}_{:2}\boldsymbol{y})u^2 + \dots$ makes it clear that the system being bilinear means having multiple autonomous linear dynamics among which to choose; moreover, a purely affine part ($\dot{\boldsymbol{y}} = \dots + \mathbf{B}\boldsymbol{u}$) can be represented by adding a dummy constant observation [27].

BDS agents: We justified the choice of BDS by saying that a bilinear map is the simplest nonlinearity[4] that can represent several sensors of interest; different values of the tensor $\mathbf{M}$ can (approximately) represent field samplers, cameras and range-finders with arbitrary intrinsic and extrinsic calibration. Thus one can design bootstrapping agents that use BDS models to represent their sensorimotor cascade that work for different sensors.

[3] Summation over repeated indices is implicit (Einstein notation).

[4] There are some analogies (but not a formal equivalence) with other models in machine learning that consider 3-way signals interactions [28] that are being studied for representing image transformations (thus related to motion).

The following is a slight modification of the agent we studied in previous work. Let $\Omega_{\boldsymbol{u}}$ be a convex set of allowable commands (modeling power constraints etc.).

Proposition 7. *Define the agent* $A_{\mathsf{BDS}}(k, n) \in \mathsf{Agents}(\mathbb{R}^k, \mathbb{R}^n)$*, whose learning phase is defined by the following set of equations. The commands* $\boldsymbol{u}$ *are chosen randomly, and three statistics* $\overline{y}^s, \mathsf{P}^{sv}, \mathsf{T}^{svi}$ *are estimated*[5]*:*

$$\mathsf{learn}: \quad \begin{cases} \boldsymbol{u} & \sim \mathcal{N}(0, \mathbf{Q}), \\ \overline{y}^s & \leftarrow \mathbb{E}\{y^s\}, \end{cases} \qquad \begin{aligned} \mathsf{P}^{sv} &\leftarrow \operatorname{cov}(y^s, y^v), \\ \mathsf{T}^{svi} &\leftarrow \mathbb{E}\{(y^s - \overline{y}^s)\, \dot{y}^v u^i\}. \end{aligned}$$

If the agent interacts with a system in $\mathsf{BDS}(n, k)$*, the following action corresponds to a servoing action to a given goal observation* $\boldsymbol{y}_\star$ *(*$\boldsymbol{y} \to \boldsymbol{y}_\star$ *locally, if the kinematics is holonomic)*[6]*:*

$$\mathsf{act}: \quad \begin{cases} \tilde{u}^i = -(y^r - y^r_\star)^* \mathsf{P}^{-1}_{rv} \mathsf{T}^{svi} \mathsf{P}^{-1}_{vq} y^q, \\ \boldsymbol{u} = \mathit{saturate}(\tilde{\boldsymbol{u}}, \Omega_{\boldsymbol{u}}). \end{cases} \tag{3}$$

Invariance properties of BDS *agents:* Note that it is not clear at all from the definition of an agent what are its assumptions about the world. Here we prove that the agent behavior is invariant to arbitrary linear transformation of input and output, as represented by the action of the GL group. This implies that there are no assumptions on the ordering of the signals, the gain of the commands, and measurements units for the observations. The first part of the proof consists in showing that the nuisances do not change the class of models; the second part shows the invariance of the agent behavior.

Proposition 8. *The* $\mathsf{BDS}(n, k)$ *family is closed with respect to the action of* $\mathsf{GL}(n)$ *on the observations and* $\mathsf{GL}(k)$ *on the commands:* $\mathsf{GL}(n) \cdot \mathsf{BDS}(n, k) \cdot \mathsf{GL}(k) \subset \mathsf{BDS}(n, k)$.

Proof: This is a simple verification that, if we let $\boldsymbol{y}' = \mathbf{A}\boldsymbol{y}$ and $\boldsymbol{u} = \mathbf{B}\boldsymbol{u}'$, (with $\mathbf{A} \in \mathsf{GL}(n)$, $\mathbf{B} \in \mathsf{GL}(k)$ invertible matrices), the relation between $\dot{\boldsymbol{y}}'$, $\boldsymbol{y}'$ and $\boldsymbol{u}'$ is bilinear. ∎

Proposition 9. *The agent* $A_{\mathsf{BDS}}(k, n)$ *is invariant to the action of* $(\mathsf{GL}(n), \mathsf{GL}(k))$ *on* $\mathsf{BDS}(n, k)$.

Proof: (sketch) The agent is minimizing the error function $J(\boldsymbol{y}) = (\boldsymbol{y} - \boldsymbol{y}_\star)^* \mathbf{P}^{-1} (\boldsymbol{y} - \boldsymbol{y}_\star)$, which is invariant to $\mathsf{GL}(n)$. In fact, we have that $\mathbf{A} \in \mathsf{GL}(n)$ maps $\boldsymbol{y} \mapsto \mathbf{A}\boldsymbol{y}$ and $\mathbf{P} \mapsto \mathbf{A}\mathbf{P}\mathbf{A}^*$ and J does not change. ∎

Note that the agent presented in [15] minimizes $\tilde{J}(\boldsymbol{y}) = \|\boldsymbol{y} - \boldsymbol{y}_\star\|^2$, which seemed to be the most intuitive choice to us, but it is in fact *not* invariant to $\mathsf{GL}(n)$.

IV. Bilinear gradient dynamics sensors

The class of *bilinear gradient dynamics sensor* (BGDS) models is a subset of BDS where the dynamics $\dot{\boldsymbol{y}}$ are assumed to depend on $\boldsymbol{y}$ itself only through the spatial gradient $\nabla \boldsymbol{y}$.

Definition 10. A system is a *bilinear gradient dynamics sensor* (BGDS), if its output is a function[7] $\boldsymbol{y}(\cdot, t) \in \mathsf{C}(\mathcal{S}; \mathbb{R})$ defined on a Riemannian manifold $\mathcal{S}$ (the *sensel space*), and the dynamics of the observations is bilinear in the gradient of

[5] Expectation represents the time average during the training phase.

[6] In 3, "saturate$(\tilde{\boldsymbol{u}}, \Omega_{\boldsymbol{u}})$" denotes the projection of $\tilde{\boldsymbol{u}}$ on the boundary of $\Omega_{\boldsymbol{u}}$.

[7] $\mathsf{C}(\mathcal{S}; \mathbb{R})$ denotes the set of smooth functions from $\mathcal{S}$ to $\mathbb{R}$.

the observations and affine in the commands. Formally, there exist two tensor fields $\mathbf{G}$ and $\mathbf{B}$ on $\mathcal{S}$ such that

$$\dot{y}(s,t) = (\mathsf{G}_i^d(s)\nabla_d y(s,t) + \mathsf{B}_i(s))u^i(t). \tag{4}$$

We denote by $\mathsf{BGDS}(\mathcal{S},k) \subset \mathcal{D}(\mathsf{C}(\mathcal{S};\mathbb{R}),\mathbb{R}^k)$ the family of all such systems with k commands and sensel space $\mathcal{S}$.

In equation (4), the symbol s represents a spatial index, the position of the sensel on the space $\mathcal{S}$, and $\nabla_d y(s)$ denotes the d-th component of the gradient with respect to s. The tensor field $\mathbf{G}$ represents the bilinear part of the dynamics, while $\mathbf{B}$ represents the purely affine part that does not depend on $\boldsymbol{y}$.

We now give the equivalent invariance properties of Proposition 8. For BDS models, we considered the effect of the linear group $\mathsf{GL}(n)$ on the observation; for BGDS, we consider the effect of diffeomorphisms of the manifold $\mathcal{S}$.

Proposition 11. *The $\mathsf{BGDS}(\mathcal{S},k)$ family is closed with respect to diffeomorphisms $\varphi \in \mathsf{Diff}(\mathcal{S})$ that act on the observations as $z(x,t) = y(\varphi(x),t)$, and the action of GL on the commands: $\mathsf{Diff}(\mathcal{S}) \cdot \mathsf{BGDS}(\mathcal{S},k) \cdot \mathsf{GL}(k) \subset \mathsf{BGDS}(\mathcal{S},k)$.*

Proof: For clarity, we prove the slightly more general result where the diffeomorphism is between two different spaces $\mathcal{S}$ and $\mathcal{Z}$:

$$\mathsf{Diff}(\mathcal{Z};\mathcal{S}) \cdot \mathsf{BGDS}(\mathcal{S},k) \cdot \mathsf{GL}(k) \subset \mathsf{BGDS}(\mathcal{Z},k).$$

Let $s \in \mathcal{S}$, $z \in \mathcal{Z}$, $s = \varphi(x)$, and $z(x,t) = y(\varphi(x),t)$. The gradients of the fields are related by

$$\nabla_d y(\varphi(x),t) = J_e^d(x)\nabla_e z(x,t) \tag{5}$$

where $\mathbf{J}$ is the jacobian of the diffeomorphism φ^{-1}. For the derivatives, we obtain:

$$\begin{aligned}
\dot{z}(x,t) &= \dot{y}(\varphi(x),t) \\
&\quad \textit{Substitution of (4).} \\
&= (\mathsf{G}_i^d(\varphi(x))\nabla_d y(\varphi(x),t) + \mathsf{B}_i(\varphi(x)))u^i(t) \\
&\quad \textit{Substitution of (5).} \\
&= (\mathsf{G}_i^d(\varphi(x))J_e^d(x)\nabla_e z(x,t) + \mathsf{B}_i(\varphi(x)))u^i(t) \\
&\triangleq (\tilde{\mathsf{G}}_i^d(x)\nabla_e z(x,t) + \tilde{\mathsf{B}}_i(x))u^i(t).
\end{aligned}$$

Therefore, after the diffeomorphism φ, the system dynamics is still bilinear, and their characteristic tensors are transformed by

$$\begin{aligned}
\tilde{\mathsf{G}}_i^d(x) &= J_e^d(x)\mathsf{G}_i^d(\varphi(x)), \\
\tilde{\mathsf{B}}_i(x) &= \mathsf{B}_i(\varphi(x)).
\end{aligned}$$

■

A. Intrinsic calibration for BGDS

An agent that wants to exploit the BGDS structure needs more information about the sensor than just the collections of measurements. In fact, in equation (4), the knowledge of the metric structure of the manifold $\mathcal{S}$ is implicit in using the gradient operator. Knowledge of the metric structure of $\mathcal{S}$ is equivalent to the *intrinsic calibration* of the sensor. In this context, the problem of intrinsic calibration is much harder than the problem as defined in the computer vision literature, where most techniques assume that the sensor geometry is known up to a few parameters to estimate and that the sensor dynamics is known (it *is* a camera). However, there is a literature of works that tackle exactly this problem [10], [19]–[21] and we can incorporate their findings in our analysis.

We model the scenario as in Fig. 2a. There is a signal $\boldsymbol{y}$ which is a function defined on the sensel space $\mathcal{S}$. This function is sampled (discretized) at n unknown points $\{s_i\}_{i=1}^n$ to give the discretized observation $y_i = y(s_i)$, which are accessible to us. We make the assumption that the sampling is dense enough with respect to the bandwidth of the signal that no information is lost, and one could reconstruct the signal from the samples. However, we do not assume that the sampling points are known, and their choice is considered a nuisance, represented by a diffeomorphism applied to a base set $\{\hat{s}_i\}_{i=1}^n$. After sampling, we insert a permutation nuisance to make it clear that the agent has no information about the ordering of the sensels.

The results in the literature can be summarized as follows. In general, it is *not* possible to reconstruct the positions $s_i \in \mathcal{S}$ of each sensel in a metrically correct way. However, it is possible to recover the correct sensel topology. One possible way to perform this is to compute the correlations $\rho_{ij} = \mathsf{corr}(y_i(t), y_j(t))$, derive logical distances $d_{ij} = \arccos(\rho_{ij})$ and then solve an embedding problem: find vectors x_i in some manifolds such that $d(x_i,x_j) \simeq d_{ij}$. The solution conserves the topology (the neighbor-neighbor relations), because if two sensels are close, their correlation is necessarily close to 1, and the embedding algorithm will place them close to each other. However, the information will *not* be recovered. This state of information can be described by a diffeomorphisms nuisance, because the action of a diffeomorphism conserves the topology, but destroys any metric information.

To summarize, the effect of sampling, permutation, calibration, and reconstruction (Fig. 2a) is equivalent to an embedding to a known space (for example $\mathbb{R}^d$, with $d \geq \dim(\mathcal{S})$) plus the action of a diffeomorphism nuisance (Fig. 2b), that, by Proposition 11, does not change the BGDS nature of the system. Thus, in designing an agent, we can assume that the sensels position in $\mathcal{S}$ is known, up to a diffeomorphism nuisance that must be taken into account.

B. Bootstrapping agents for BGDS

Definition 12. Assume that, possibly after a calibration stage, the Riemannian structure of the manifold $\mathcal{S}$ is known (up to a diffeomorphism). Define the agent $A_{\mathsf{BGDS}}(k,\mathcal{S}) \in \mathsf{Agents}(\mathsf{C}(\mathcal{S};\mathbb{R}),\mathbb{R}^k)$, whose learning stage consists in choosing random commands $\boldsymbol{u}$, and learning the three tensor fields $\mathbf{R}(s), \mathbf{C}(s), \mathbf{H}(s)$ on $\mathcal{S}$ from statistics of the data:

$$\mathsf{learn}: \begin{cases} \boldsymbol{u} \sim \mathcal{N}(0,\mathbf{Q}), \\ \mathsf{R}_{ef}(s) \leftarrow \mathbb{E}\{\nabla_e y(s)\nabla_f y(s)\}, \\ \mathsf{H}_d^i(s) \leftarrow \mathbb{E}\{\dot{y}(s)\nabla_d y(s)u^i\}, \\ \mathsf{C}^i(s) \leftarrow \mathbb{E}\{\dot{y}(s)u^i\}. \end{cases} \tag{6}$$

Here $\mathbf{R}(s)$ is the covariance of the gradient of the signal (a square matrix at each point of the manifold $\mathcal{S}$); $\mathbf{C}(s)$ and $\mathbf{H}(s)$ are proxies for the bilinear and affine part of the dynamics.

The first example of behavior for this agent is servoing to a goal observation $\boldsymbol{y}_\star$, in analogy to the corresponding behavior of the BDS agent.

Proposition 13. *(Servoing) The servoing strategy analogous to equation (3) that makes $\boldsymbol{y} \to \boldsymbol{y}_\star$ (locally) is*[8]

$$\mathsf{act}: \begin{cases} \tilde{u}^i = -\mathsf{Q}_{ij} \int \underbrace{(y(s) - y_\star(s))}_{error} \times \\ \qquad \times \underbrace{\left[\mathsf{H}^i_d(s)\mathsf{R}^{dD}\nabla_D y(s) + \mathsf{C}^i(s)\right]}_{jacobian} \underbrace{\sqrt{\det(\mathbf{R}(s))}\, d\mathcal{S}}_{normalized\ measure}, \\ \boldsymbol{u} = saturate(\tilde{\boldsymbol{u}}, \Omega_{\boldsymbol{u}}). \end{cases} \tag{7}$$

Proof: (sketch) The learned tensors converge to

$$\mathsf{H}^i_d(s) = \mathsf{G}^D_j(s)\mathsf{R}_{Dd}(s)\mathsf{Q}^{ij}, \qquad \mathsf{C}^i(s) = \mathsf{B}_j(s)\mathsf{Q}^{ij}. \tag{8}$$

As for the case of the BDS agent, the control command can be derived as gradient descent for the error function

$$J(\boldsymbol{y}) = \tfrac{1}{2}\int (y(s) - y_\star(s))^2 \sqrt{\det(\mathbf{R}(s))}\, d\mathcal{S}. \tag{9}$$

The time derivative is $\dot{J}(\boldsymbol{y}) = [\int (y(s) - y_\star(s))\sqrt{\det(\mathbf{R}(s))}(\mathsf{G}^d_i(s)\nabla_d y(s) + \mathsf{B}_i(s))d\mathcal{S}]\dot{u}^i$. Some algebra and equations (8) show that the descent direction can be written as in (7). ∎

The second example is an anomaly detector, which detects changes in the observations that cannot be explained by the agent's own motion. This is the passive task that we will demonstrate in the experiments section.

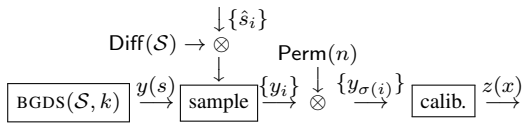
Proposition 14. *(Prediction/anomaly detection) We can define a signal $\hat{\dot{y}}(s)$ which predicts $\dot{y}(s)$ as:*

$$\hat{\dot{y}}(s) = \left[(\mathsf{H}^i_d(s)\mathsf{R}^{dD}(s)\nabla_D y(s) + \mathsf{C}^i(s))\right] \mathsf{Q}_{ij}\dot{\boldsymbol{u}}^j, \tag{10}$$

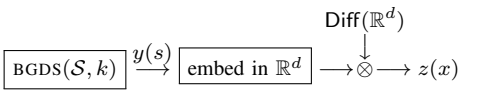
and an anomaly detection signal can be defined as

$$d(s) = \max\{-\hat{\dot{y}}(s)\dot{y}(s), 0\}. \tag{11}$$

[8]In these equations, following the usual tensor notation, raising both indices corresponds to inverting the matrix: $\mathsf{R}^{dD}(s) \triangleq (\mathsf{R}(s)^{-1})^{dD}$.



(a) Reference model for the unknown sampling process.



(b) Equivalent model with single diffeomorphism nuisance.

Figure 2. *Intrinsic calibration of* BGDS *as reduction of group nuisances*. In BGDS systems, the observations are a function defined on some manifold $\mathcal{S}$. (a) At the beginning, we only have the uninterpreted data streams $\{y_i\}$ but we do not know the position of each sensel $s_i \in \mathcal{S}$. We can model this state of information by assuming that a fixed sensel configuration $\{\hat{s}_i\}$ is perturbed by an unknown diffeomorphism $\mathsf{Diff}(\mathcal{S})$, followed by a permutation that scrambles the sensels. Previous research has shown that it is possible to design calibration procedure that reconstruct the *topology* of the sensels from the raw data streams, while the *metric* information is not available. This situation can be modeled as in (b): knowing the topology but not the metric information means that there is a diffeomorphism nuisance. Therefore, with a calibration procedure we can reduce complete ignorance of the sensels position to "just" a diffeomorphism nuisance.

Equation (10) is simply the prediction given the learned model: equation (4) written using the learned tensors. The detector (11) returns a positive response when the predicted and observed derivative disagree on their sign; if the actual sensors is precisely a BGDS, this signal detects extraneous objects on the field of view.

Invariance properties: In complete analogy with Proposition 9 for the BDS agent, the BGDS agent is invariant to diffeomorphisms of the sensel space.

Proposition 15. *The action (7) is invariant to the actions of $(\mathsf{Diff}(\mathcal{S}), \mathsf{GL}(k))$ on $\mathsf{BGDS}(\mathcal{S}, k)$.*

Proof: (sketch) This is achieved by the normalization factor $\sqrt{\det(\mathbf{R}(s))}$; recall that $\mathbf{R}(s)$ is the covariance of $\nabla y(s)$: if the image is stretched, the covariance of the gradient decreases, compensating for a large area. More formally, under a diffeomorphism $\varphi \in \mathsf{Diff}(\mathcal{S})$ such that $s = \varphi(x)$ and $\mathbf{J} = \partial\varphi/\partial x$, the volume form undergoes the transformation $d\mathcal{S} \mapsto |\det(\mathbf{J})|d\mathcal{S}$, while the new covariance of the gradient is $\mathbf{R}(x) = \mathbf{J}\mathbf{R}(\varphi(x))\mathbf{J}^*$. These observations (and a bit of algebra) imply that the integral in equation (9) is equivalent if written for the coordinates x or $s = \varphi(x)$. ∎

Comparison between BGDS *and* BDS*:* The BGDS agent appears much more complicated than the BDS agent, and works on a subset of the models (BGDS $\subset$ BDS); however, there is a great efficiency gain. Let d be the dimension of the space $\mathcal{S}$. Then the tensor field $\mathbf{H}$ can be represented by an $d \times k$ matrix at each point of $\mathcal{S}$. If $\mathcal{S}$ is discretized to n sensels, then $\mathbf{H}$ has a total of $n \times d \times k$ elements and $\mathbf{C}$ has $n \times k$ elements. Thus, as shown in Table III, learning of a BGDS requires $\mathcal{O}(n)$ storage versus $\mathcal{O}(n^2)$ for a BDS.

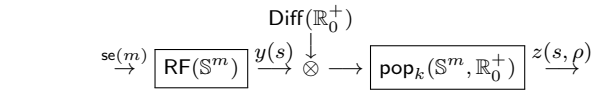
Table III
STORAGE REQUIREMENTS FOR THE BDS AND BGDS AGENTS

$A_{\mathsf{BDS}}(n,k)$			$A_{\mathsf{BGDS}}(n,k)$	
$\mathbf{P}$	$\mathcal{O}(n^2)$	statistics	$\mathbf{R}(s_i)$	$\mathcal{O}(nd^2)$
$\mathbf{T}$	$\mathcal{O}(n^2k)$	dynamics	$\mathbf{C}(s_i)$	$\mathcal{O}(nk)$
			$\mathbf{H}(s_i)$	$\mathcal{O}(nkd)$

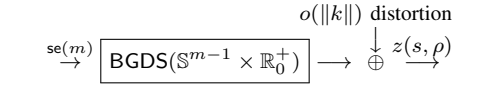
n: number of sensels; k: number of commands; d: dimension of $\mathcal{S}$.

V. BGDS APPROXIMATIONS FOR RANGE-FINDERS

We have been considering three canonical sensors: the field sampler, the range-finder, and a camera. In previous work, we have shown that the BDS model was a good but not perfect model for all of them. In particular, range-finders



(a) Series of range-finder, unknown distortion, and population code .



(b) Equivalent model: BGDS with unstructured (i.e. non-group) nuisance.

Figure 3. A range-finder + population code can be approximated by a BGDS with low-distortion (Proposition 17).

are not BDS/BGDS because of the nonlinearity in the dynamics (Table I). Here we show that a range finder model can be represented with small distortion as a BGDS, if the data is first preprocessed by a transformation similar to a population code.

Definition 16. Consider a signal y defined on a domain $\mathcal{S}$ ($y : \mathcal{S} \to \mathcal{O}$), and a symmetric kernel $\psi : \mathcal{O} \times \mathcal{O} \to \mathbb{R}$. Then we call "population code" the signal $z : \mathcal{S} \times \mathcal{O} \to \mathbb{R}$ defined by $z(s, x) = \psi(x, y(s))$. We define a stateless dynamical system computing the population code as $\mathsf{pop}_\psi(\mathcal{S}, \mathcal{O})$.

For type-checking, we note that $\mathsf{pop}_\psi(\mathcal{S}, \mathcal{O}) \in \mathcal{D}(\mathsf{C}(\mathcal{S} \times \mathcal{O}; \mathbb{R}), \mathsf{C}(\mathcal{S}; \mathcal{O}))$: it augments the dimension of the observations from a field defined on $\mathcal{S}$ to a field defined on $\mathcal{S} \times \mathcal{O}$.

After this transformation, the dynamics appear simplified. Let $\mathsf{RF}(\mathbb{S}^m, k)$ be the family of all range-finders models for robots in $\mathsf{SE}(m)$ with k linear/angular velocity commands in $\mathsf{se}(m)$. The output of a rangefinder is a function defined on the sphere $\mathbb{S}^m$ to $\mathbb{R}_0^+$, therefore the family $\mathsf{RF}(\mathbb{S}^m, k)$ is a subset of $\mathcal{D}(\mathsf{C}(\mathbb{S}^m; \mathbb{R}_0^+), k)$.

Proposition 17. *The dynamics of a range-finder whose output is filtered by a population code can be represented with small distortion by a* BGDS*, except near occlusions, with a small distortion bounded by $\|\psi\|$:*

$$\mathsf{pop}_\psi(\mathbb{S}^m, \mathbb{R}_0^+) \cdot \mathsf{RF}(\mathbb{S}^m, k) \subset \mathsf{BGDS}(\mathbb{S}^m \times \mathbb{R}_0^+, k) \oplus o(\|\psi\|).$$

Proof: For simplicity, we can picture the situation in the plane ($m = 2$), but everything is valid in 3D as well. Let $O \subset \mathbb{R}^2$ be the subset of obstacles in the plane that are opaque to the range-finder. Define an indicator function $\delta_O : \mathbb{R}^2 \to \{0, 1\}$ such that $\delta_O(x)$ is 1 if the world is opaque at $x \in \mathbb{R}^2$. We define a field sampler sampling the field δ_O as follows:

$$\begin{aligned} f : \mathbb{R}^2 &\to \mathbb{R} \\ \boldsymbol{q} &\mapsto \delta_O(\mathbf{R}\boldsymbol{q} + \boldsymbol{t}), \end{aligned} \tag{12}$$

where $\boldsymbol{q} \in \mathbb{R}^2$ is the sensel position in robot frame, and $(\mathbf{R}, \boldsymbol{t}) \in \mathsf{SE}(2)$ is the robot pose. As previously discussed, every field sampler is a BGDS.

Let $y : \mathbb{S}^1 \to \mathbb{R}_0^+$ be the output of a range-finder. Define its population code representation with a delta kernel ($k(x, y) = \delta_x(y)$) as $z(s, \rho) = \psi(\rho, y(s)) = \delta_{y(s)}(\rho)$. Note that in the population code for a range finder $s \in \mathbb{S}^1$ ranges over direction and $x \in \mathbb{R}_0^+$ ranges over distances; therefore, $z(s, \rho)$ is 1 if the obstacle in direction s is at distance x so it represents a local polar map of the environment. Define the polar-to-cartesian change of coordinates $\varphi : (s, \rho) \mapsto (\rho\cos(s), \rho\sin(s))$, which is a diffeomorphism except at the origin. The function z can be expressed as a function of the field sampler 12 as $z(s, \rho) = f(\varphi(s, \rho))$. This means that its output is a diffeomorphism of the output of a BGDS model.

Note, however, that so far we used a delta kernel $\psi = \delta$ and an indicator function δ_O for the obstacle set; this makes the output of the sensor sparse and discontinuous, which makes it impossible to represent the dynamics with partial differential equations. The solution is to use a kernel ψ with some smoothing (a trick used for different reasons in SLAM, leading to "relaxed" likelihood models), to which it (approximately) corresponds a smoothing of the field δ_O. The math only works in the limit as $\psi \to \delta$, therefore there is a bounded small distortion. ∎

As a consequence, we know that the BGDS agent can learn, with small approximation, the dynamics of a range finder. We can take this further: suppose that we have a sensor whose output is an unknown function of the range; that is, instead of the ranges y_i, we have measurements $y_i' = f(y_i)$ for an unknown $f \in \mathsf{Diff}(\mathbb{R}_0^+)$. We can show that this transformation would be tolerated by the agent, because the resulting system is still a BGDS. In summary, it is possible to get the data from a range finder, distort it by a nonlinear function, shuffle the measurements, and then apply one of the calibration techniques that recover the topology, and the BGDS agent behavior will be invariant to all those group nuisances.

VI. Experiments

This section shows that BGDS bootstrapping agents can deal with the sensor suite of real-world robotic platforms with heterogeneous sensors. After minimal preprocessing, heterogeneous data from range-finders and cameras can be handled by exactly the same algorithm, with no other prior information. In [15], we evaluated BDS models on servoing. Here, we consider instead a passive task that can be tested on logged data. The task is anomaly detection: the agent must discover which changes in its stimuli can be explained by its own motion, and which are due to independent causes (e.g. objects moving in the field of view).

Datasets: We use data available from the Rawseeds project [29]. The robot is a differential drive platform with a full sensor suite onboard: 2 Sick range-finders (180 deg field of view, 181 readings, 70 Hz, max range of 80 m), 2 Hokuyo range-finders (270 deg field of view, 626 readings each, 10Hz, max range of 6 m), one omnidirectional camera (RGB 640x640, 15fps), one frontal camera with fisheye lens (RGB 640x480, 30fps), and a Point Gray Triclops (grayscale, 3x320x240, 15fps), GPS and IMU. The available logs correspond to a few hours of operation both indoors and outdoors, for a total of about 500GB of uncompressed data.

The commands $\boldsymbol{u}$ are the linear and angular velocities obtained from differentiating the odometry data. We set $u^0 = v$ (linear) and $u^1 = \omega$ (angular) — this is mainly for having clearer pictures, because we proved that the final result is invariant to any linear transformation.

Tests with range-finder data: For range-finder data, we demonstrate the bootstrapping pipeline as explained in Section V, formalized in Fig. 3, and cartoonishly represented in Fig. 4a. We take the data from the two Sick range finders, mounted with a heading of approximately 0 and 180deg with respect to the robot. The raw readings are deformed by a nonlinear function $x \mapsto 1/x$ (nearness instead of range) and shuffled according to a random permutation. Thus the initial input to the pipeline is a set of 362 shuffled, warped sensels values $\{y_i\}_{i=1}^n$. An embedding algorithm [21] recovers the position of the sensels on the unit circle using a metric obtained by computing the information distances of the sensels. The reconstruction is accurate up to a diffeomorphism nuisance: that is, if $\theta_i \in \mathbb{S}^1$ is the real angle, we can estimate

$\tilde{\theta}_i = \varphi(\theta_i)$ for $\varphi \in \mathsf{Diff}(\mathbb{S}^1)$. In practice, this means that we can reconstruct only the order of the sensels on the circle. We normalize each sensel value in the range $[0, 100]$ by computing the percentile of $y_i(t)$ to the whole history $y_i[-\infty, t]$; this normalizes the nonlinear scaling. Then we apply a population code filter with a small kernel ($\psi = 0.5$ against the $[0, 100]$ range). In the end, from shuffled distorted values $\{y_i\}_{i=1}^n$, we have obtained a 2D field $y(\tilde{\theta}, \tilde{\rho})$, where $\tilde{\theta} \in [0, 2\pi]$ is a (warped) angle and $\tilde{\rho} \in [0, 100]$ is a nonlinear function of the range. This "image" is diffeomorphic to a polar map of the environment; but notice that this is obtained not from prior geometric knowledge but only as the consequence of a data-agnostic pipeline.

In this case, the sensel space $\mathcal{S}$ ($\mathcal{S} = [0, 2\pi] \times [0, 100]$) has dimension $d = 2$ and there are $k = 2$ commands (linear and angular velocity). Therefore the tensor $\mathbf{H}$ has 4 components, each of which is a field across the sensel space. The tensor $\mathbf{C}$ has 2 components (for linear and angular velocity), but we do not show it for the camera data, because it is in theory 0, and in practice insignificant noise. The 4 components of the tensor field $\mathbf{H}$ are shown in Fig. 5 as 4 false color images (white: zero, red: positive, blue: negative). Refer to the caption for some interpretation, not always intuitive. An example result of anomaly detection is shown in Fig. 7e. This is done by using the learned tensors to compute the prediction signal (10), and then computing the detection signal (11). The figure shows the total anomaly reported per sensel over time. Anomalies are reported constantly for the sensor horizons (front and back) where objects pop into view; the occasional traces correspond to people walking around the robot.

Tests with camera data: We stitched together the images from the omnidirectional camera, the frontal camera, and one of the stereo triplets to obtain a unique 640x480 frame from which we compute a grayscale signal (Fig 4b). In this case, we start from the knowledge of the correct sensel topology given from the raw images. However, there is still a diffeomorphism nuisance, because we assume no prior information on the intrinsic calibration of the cameras (the unknown diffeomorphism is the one—actually, there are three—that maps each pixel in the composite frame to the corresponding direction on the visual sphere).

The source image has three components (R,G,B). One must choose how to convert the raw RGB signal into a scalar quantity. In this paper, we are more interested in the general issues which are common across sensory modalities rather than specific issues of a particular sensor. We considered two filters: 1) a standard RGB to grayscale (luminance) conversion, and 2) grayscale, followed by the computation of the image contrast: $y(s) \mapsto \|\nabla y(s)\|$. We found that the results seem to be robust to the preprocessing step and should be largely invariant for any other local filter applied to the images.

Before looking at the learning results, it is instructive to look at the first-order data statistics, such as the mean and variances of the signals (Fig. 8). Already these simple statistics show that parts of the image are non informative: the borders of the omnidirectional camera and the camera reflection in the conic mirror have almost zero variance. One could use this information to ignore those parts; however we shall see that the tensor learning will ignore non informative parts automatically.

Fig. 6 shows the learning results, using the grayscale signal. Also in this case the sensel space $\mathcal{S} = [1, 640] \times [1, 480]$ has dimension $d = 2$ and there are $k = 2$ commands. Therefore, the tensor $\mathbf{H}$ has 4 components, each of which is a field across the sensel space, that can be displayed as 4 false colors images. Interpreting those images is not immediate. Remember that, ultimately, the tensor $\mathbf{H}_d^i$ shows how the derivative $\dot{y}$ is affected by the intensity of the d-th component of $\nabla_d y$ (∇_0 being the horizontal gradient and ∇_1 being the vertical gradient). Note that all fixed parts of the robot reflected in the mirror appear as white (zero values). In the field $\mathbf{H}_1^0$ (corresponding to linear velocity v and vertical gradient), it is easy to understand why the ceiling appears red (negative) and the floor blue (positive): if the robot moves with $v > 0$, and there is a positive gradient $\nabla_1 y > 0$, then one expects $\dot{y} > 0$ in the ceiling and $\dot{y} < 0$ for the floor. The part of the omnidirectional camera is all red (instead of blue) because the image is mirrored. The tensor $\mathbf{H}$ contains both intrinsic information about the sensor (direction of pixels) and extrinsic information (interaction sensor-command), as well as statistics of the environment (things on average further away correspond to lower response to translational motion).

One could make similar interpretations using the concept of *optic flow* (apparent retinal motion), however, notice we never compute optic flow and we do not assume the agent has the semantics of "motion" at all; we only use the quantities $\dot{y}$ and ∇y that can be computed directly from the data without problems of regularizations. Roberts *et al.* [30] presents an analogous approach to learning "optic flow subspaces" using feature tracking and optimization.

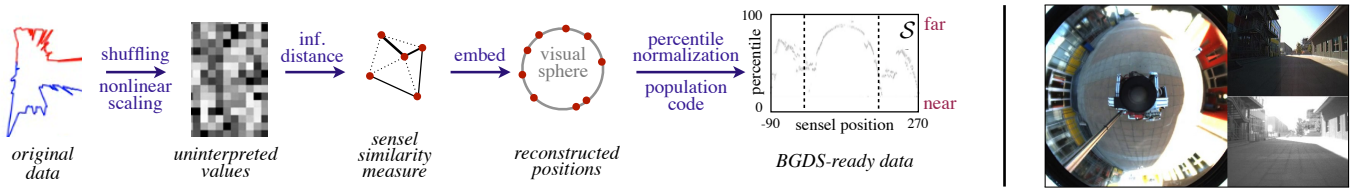
Fig. 9 shows the analogous results using the contrast signal instead of grayscale. The learned tensors are very similar; the results should be invariant to all local image operations.

Fig. 7a–7d show an example of anomaly detection. As expected, objects (such as people) that move not coherently with the motion are readily detected. But the model also breaks down at occlusions and for very fast rotational motions; that is, where the signal evolution cannot be represented accurately using a smooth model such as (4). We also found false positive responses for finer details (Fig. 10).

Supplementary materials: The website http://purl.org/censi/2011/bgds contains several videos showing the original data, the learning process, prediction, and detection.

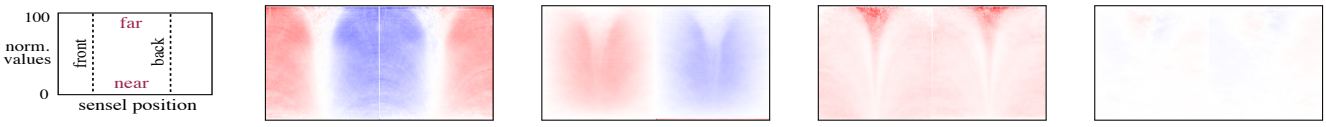
VII. Conclusions

This paper presented a formalization of bootstrapping as a problem of rejection of group nuisances on the representation of observations and commands. This formalization identifies a falsifiable property for bootstrapping agents, and allows modular analysis of data processing stages by the nuisances they equalize or add. We introduced BGDS models, and relative learning agents, as an efficient specialization of BDS models used in previous work. We studied the invariance of BGDS agents against groups nuisances, such as linear transformations of the commands and diffeomorphisms of the sensel space. A BGDS agent needs knowledge of the metric structure of



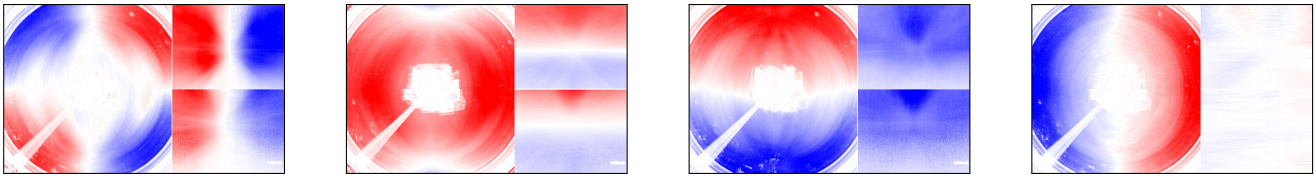
(a) Bootstrapping pipeline for range-finder data (b) Composite camera frame

Figure 4. *Data sources used in the experiments.* (a) For range-finder data, we apply the complete bootstrapping pipeline as described in Section V: we start from shuffled and distorted measurements; we compute sensels similarities with the information distance between the sensels values, from which an embedding on the visual sphere (circle) can be obtained [21]; finally, population coding (Definition 16) is used to obtain a two-dimensional "image" which is diffeomorphic to a polar map of the immediate surroundings. (b) For camera data, we use a composite frame obtained by stitching together the frames from an omnidirectional camera, a wide-angle frontal camera, and a gray-scale camera part of a stereo triplet. No previous knowledge of the optics is used.



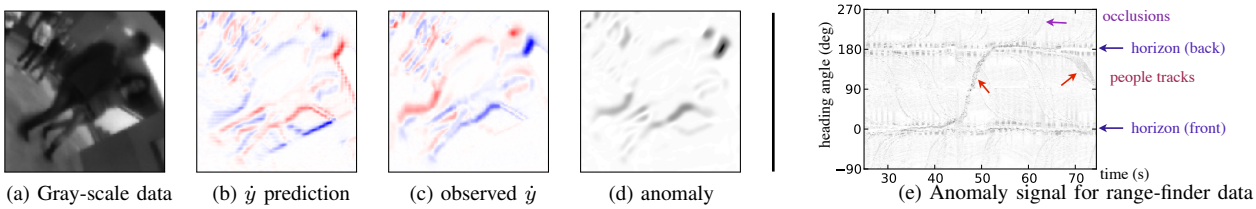
(a) $\mathbf{H}_0^0$ (hor. gradient and v) (b) $\mathbf{H}_1^0$ (ver. gradient and v) (c) $\mathbf{H}_0^1$ (hor. gradient and ω) (d) $\mathbf{H}_1^1$ (ver. gradient and ω)

Figure 5. *Tensors learned from range-finder data, with population code processing.* To understand these figures, it helps to think that the population code representation of range-finder data is diffeomorphic to a polar map of the environment. On the x-axis we have the angle (up to a diffeomorphism); on the y-axis we have the normalized values of the readings in percentiles (i.e., distance up to a diffeomorphism). In (c)-(d) the tensors that represent the interaction between angular velocity and horizontal and vertical gradient are respectively a constant and zero because the effect of rotation is just to translate horizontally these diagrams. (a)-(b) are not of easy interpretation, but the (anti) symmetry between the frontal and back range-finders is evident.



(a) $\mathbf{H}_0^0$ (hor. gradient and v) (b) $\mathbf{H}_1^0$ (ver. gradient and v) (c) $\mathbf{H}_0^1$ (hor. gradient and ω) (d) $\mathbf{H}_1^1$ (ver. gradient and ω)

Figure 6. *Tensors learned from camera data* (white: zero, red: positive, blue: negative). The tensor $\mathbf{H}$ encodes the sensors intrinsic and extrinsic calibration, along with some ancillary statistics about the environment. The interpretation is not immediate. Note first that the parts that are not influenced by the robot motion (reflected camera, borders) appear as white, meaning that the tensor there is zero and negligible. The intensity depends on the average distance to the obstacle and to the pixel density on the visual sphere. The color depends on the pixel orientation on the visual sphere: see the text for more discussion.



(a) Gray-scale data (b) $\dot{y}$ prediction (c) observed $\dot{y}$ (d) anomaly (e) Anomaly signal for range-finder data

Figure 7. *Anomaly detection using learned models.* We test the models on the task of anomaly detection, a passive task that can be done on logged data. The learned tensors are used to predict $\dot{y}$ using (10), and compute the anomaly detection signal (11). (a)–(d) show the results on camera data on a single frame. (e) shows the results for range finder data; the anomaly detection signal is shown over time (x axis) for each sensel (y axis). The constant traces represent the sensors horizon, where the model cannot predict the observations. The other traces represent people walking past the robot or in opposite direction. Occlusions (e.g., walking past open doors) also are detected as anomalies as the continuous model (4) cannot represent the discontinuous dynamics.

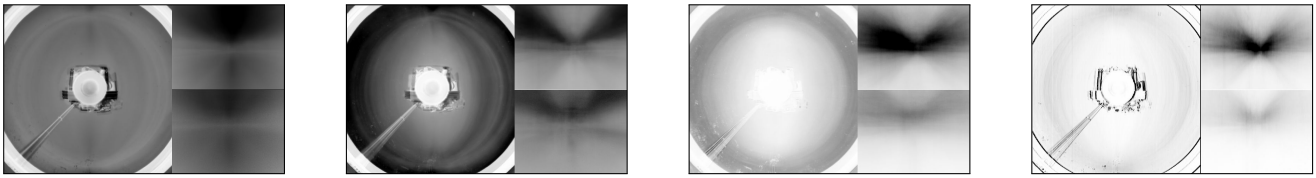
the observation space, but the invariance to diffeomorphisms ensures that it can use the imperfect calibration obtained by the generic calibration algorithms in the literature. We tried these models on heterogenous real-world data from publicly available datasets. Exactly the same algorithm can use data from different sensors (camera, range-finder) to solve the same task (anomaly detection).

Future work will entail solving higher-level tasks (fault detection [31], exploration, mapping) based on these models, extending them such that they can represent more complex kinematics and non-smooth phenomena such as occlusions [17]; and, in general, understand whether this provable, falsifiable approach to bootstrapping can be extended to higher-level cognitive abilities.

References

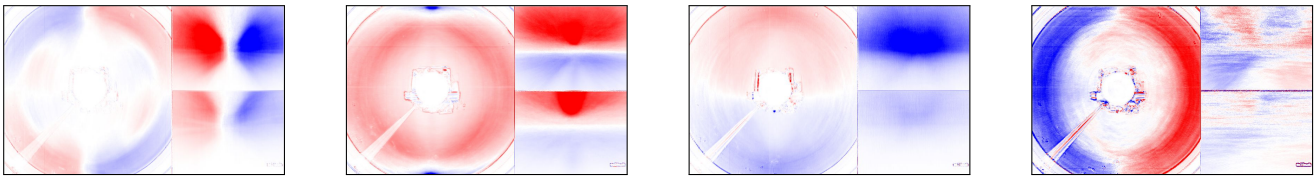
[1] Cohen *et al.*, "Functional relevance of cross-modal plasticity in blind humans," *Nature*, vol. 389, no. 6647, 1997. DOI.

[2] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, 2006. DOI.



(a) $\mathbb{E}\{\boldsymbol{y}\}$ (b) var$\boldsymbol{y}$ (c) var$\dot{\boldsymbol{y}}$ (d) var$\nabla_1\boldsymbol{y}$)

Figure 8. *Statistics for the camera data (gray-scale signal).* In these figures we adopt the convention that white=zero and darker=positive. Subfigure (a) shows the observations mean. It is possible to see that the camera reflection remains fixed in the field of view, while the rest appears as a blur. Subfigure (b) shows the data variance. The part comprising the camera reflection appears as white, meaning that the variance is very small. Subfigure (c) shows the variance of $\dot{\boldsymbol{y}}$. This shows what parts of the image have the faster dynamics: apparently, more things happen in the distance rather than in the vicinity of the robot. Subfigure (d) shows the variance of $\nabla_1\boldsymbol{y}$. It shows that some gradients are usually very strong, for example at the border between the subimages. Assuming that the world has uniform texture energy spectrum over the field of view, the energy of $\nabla\boldsymbol{y}$ is increasing with the distance, because the power spectrum of ∇y increases as $\boldsymbol{y}$ is zoomed out.



(a) $\mathbf{H}_0^0$ (hor. gradient and v) (b) $\mathbf{H}_1^0$ (ver. gradient and v) (c) $\mathbf{H}_0^1$ (hor. gradient and ω) (d) $\mathbf{H}_1^1$ (ver. gradient and ω)

Figure 9. *Tensors learned from camera data, after contrast operation* (white: zero, red: positive, blue: negative). These are the result equivalent to those in Fig. 6 with different set of filters: instead of the grayscale signal we use *contrast*, and we take the sign of temporal and spatial derivatives in (6).

(a) Gray-scale data (b) $\dot{y}$ prediction (c) observed $\dot{y}$ (d) anomaly

Figure 10. *Example of failure for very fine details.* We noticed that the detection would give false positives for very fine textures. This figure shows a blowup of a 16px×16px detail of a door frame. The observed derivative and the prediction are apparently a very close match, but a closer examination reveals that the prediction is shifted 1 pixel to the left. Subfigure (d) shows the detection signal: because of the 1 pixel inconsistency, two bands are detected as inconsistent. We do not have a full explanation of this phenomenon, but probably it is linked to the fact that we approximate the derivative at instant k using the the difference $y_k - y_{k-1}$, thus introducing a phase delay.

[3] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, 2009. DOI.

[4] D. George and J. Hawkins, "Towards a mathematical theory of cortical micro-circuits," *PLoS Comput Biol*, vol. 5, no. 10, 2009. DOI.

[5] M. Lungarella, G. Metta, R. Pfeifer, and G. Sandini, "Developmental robotics: a survey," *Connection Science*, vol. 15, 2003. DOI.

[6] Asada *et al*, "Cognitive developmental robotics: A survey," *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 1, 2009. DOI.

[7] D. Pierce and B. Kuipers, "Map learning with uninterpreted sensors and effectors," *Artificial Intelligence*, vol. 92, no. 1-2, 1997. DOI.

[8] J. Provost and B. Kuipers, "Self-organizing distinctive state abstraction using options," in *Proceedings of the International Conference on Epigenetic Robotics (EpiRob)*, 2007. (link).

[9] J. Stober and B. Kuipers, "From pixels to policies: A bootstrapping agent," in *Proceedings of the International Conference on Development and Learning (ICDL)*, 2008. DOI.

[10] J. Modayil, "Discovering sensor space: Constructing spatial embeddings that explain sensor correlations," in *Proceedings of the International Conference on Development and Learning (ICDL)*, 2010. DOI.

[11] R. S. Sutton, H. R. Maei, D. Precup, S. Bhatnagar, D. Silver, C. Szepesvari, and E. Wiewiora, "Fast gradient-descent methods for temporal-difference learning with linear function approximation," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2009. DOI.

[12] M. Hutter, *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Berlin: Springer, 2004. (link).

[13] J. Bongard, V. Zykov, and H. Lipson, "Resilient Machines Through Continuous Self-Modeling," *Science*, vol. 314, no. 5802, 2006. DOI.

[14] D. Nguyen-Tuong and J. Peters, "Using model knowledge for learning inverse dynamics," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010. DOI.

[15] A. Censi and R. M. Murray, "Bootstrapping bilinear models of robotic sensorimotor cascades," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011. To appear. (link).

[16] J.-S. Gutmann, G. Brisson, E. Eade, P. Fong, and M. Munich, "Vector field SLAM," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010. DOI.

[17] S. Soatto, "Steps towards a theory of visual information2010," Tech. Rep. 100028, UCLA-CSD, 2010. (link).

[18] G. S. Chirikjian, "Information theory on lie groups and mobile robotics applications," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010. DOI.

[19] J. Stober, L. Fishgold, and B. Kuipers, "Sensor map discovery for developing robots," in *AAAI Fall Symposium on Manifold Learning and Its Applications*, 2009. (link).

[20] M. Boerlin, T. Delbruck, and K. Eng, "Getting to know your neighbors: unsupervised learning of topography from real-world, event-based input," *Neural computation*, vol. 21, no. 1, 2009. (link).

[21] E. Grossmann, J. A. Gaspar, and F. Orabona, "Discrete camera calibration from pixel streams," *Computer Vision and Image Understanding*, vol. 114, no. 2, 2010. DOI.

[22] J. Rothman, *An introduction to the theory of groups*. Springer-Verlag, 1995.

[23] V. Varadarajan, *Lie Groups, Lie Algebras, and Their Representation*. Springer, 1984.

[24] M. do Carmo, *Riemannian Geometry*. Birkhauser, 1994.

[25] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, tensor analysis, and applications*, vol. 75 of *Applied Mathematical Sciences*. New York: Springer-Verlag, second ed., 1988. (link).

[26] S. Singh and M. James, "Predictive state representations: A new theory for modeling dynamical systems," in *International Conference on Uncertainty in Artificial Intelligence*, 2004. (link).

[27] D. L. Elliott, *Bilinear control systems: matrices in action*. Springer, 2009. DOI.

[28] R. Memisevic and G. Hinton, "Learning to represent spatial transformations with factored higher-order boltzmann machines," *Neural Computation*, vol. 22, no. 6, 2010. DOI.

[29] Ceriani *et al.*, "Rawseeds ground truth collection systems for indoor self-localization and mapping," *Autonomous Robots*, vol. 27, no. 4, 2009. DOI.

[30] R. Roberts, C. Potthast, and F. Dellaert, "Learning general optical flow subspaces for egomotion estimation and detection of motion anomalies," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. DOI.

[31] Z. Kira, "Modeling cross-sensory and sensorimotor correlations to detect and localize faults in mobile robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007. DOI.